

1. (Amended) A method for precise feedback data generation and updating during compile-time optimizations, within an optimizing compiler, comprising:

- Sub B17  
A2
- (1) accessing a first intermediate representation of source code of a computer program, wherein said first intermediate representation includes instructions instrumented into the source code of said computer program;
  - (2) annotating said first intermediate representation with previously-gathered feedback data from a plurality of sample executions of said computer program;
  - (3) updating said feedback data during compile-time according to a pre-defined propagation scheme;
  - (4) performing an optimization of said first intermediate representation annotated with said feedback data updated in step (3), thereby producing a transformed intermediate representation; and
  - (5) repeating steps (3) and (4) at least once;

whereby the compiler produces more efficient executable program code from said first intermediate representation, thus speeding up execution of said computer program.

Sub B17  
A3

6. (Amended) A computer program product comprising a computer usable medium having computer readable program code means embodied in said medium for causing an application program to execute on a computer that performs precise feedback data generation and updating during compile-time optimizations, within an optimizing compiler, said computer readable program code means comprising:

Sub 27  
D1

first computer readable program code means for causing the computer to access a first intermediate representation of source code of a computer program, wherein said first intermediate representation includes instructions instrumented into the source code of said computer program;

second computer readable program code means for causing the computer to annotate said first intermediate representation with previously-gathered feedback data from a plurality of sample executions of said computer program;

third computer readable program code means for causing the computer to update said feedback data during compile-time according to a pre-defined propagation scheme;

fourth computer readable program code means for causing the computer to perform an optimization of said first intermediate representation annotated with said feedback data updated by said third computer readable program code means, thereby producing a transformed intermediate representation; and

fifth computer readable program code means for causing the computer to re-execute said third and fourth computer readable program code means at least once;

whereby the compiler produces more efficient executable program code from said first intermediate representation, thus speeding up execution of said computer program.

Please add the following new claims 9-20:

- Sub B<sup>3</sup> 7
9. (New) A method for compile-time optimization comprising:
- (1) accessing a first intermediate representation of source code of a computer program, wherein the first intermediate representation includes instructions instrumented into the source code;
  - (2) annotating the first intermediate representation with previously-gathered global and local frequency data from a plurality of sample executions of the computer program;
  - (3) updating the global and local frequency data according to a pre-defined propagation scheme;
  - (4) performing an optimization of the first intermediate representation annotated with the global and local frequency data updated in step (3) to produce a transformed intermediate representation; and
  - (5) repeating steps (3) and (4) at least once.
- A

- Sub D<sup>1</sup> 10.
- The method of claim 9, wherein step (4) comprises the step of performing at least one of the following optimizations:
- (i) dead code elimination;
  - (ii) dead store elimination;
  - (iii) branch elimination; and
  - (iv) code transformation.

11. The method of claim 9, wherein the first intermediate representation is a tree corresponding to a procedure within the source code.

12. The method of claim 11, wherein step (2) comprises the steps of:

- (a) constructing a control flow graph from the tree; and
- (b) annotating a global or local frequency value of the global and local frequency data to an edge of the control flow graph, wherein the global or local frequency value corresponds to the number of times that the edge was traversed during the plurality of sample executions of the computer program.

13. The method of claim 12, wherein the global and local frequency value annotated to the edge of the control flow graph is one of the following:

- (i) EXACT;
- (ii) GUESS;
- (iii) UNKNOWN;
- (iv) UNINIT; and
- (v) ERROR.

14. (New) A method for compile-time optimization comprising:

- (1) accessing a first intermediate representation of source code of a computer program, wherein the first intermediate representation includes instructions instrumented into the source code;

Sub B<sup>4</sup> 7

- (2) annotating the first intermediate representation with previously-gathered feedback data from a plurality of sample executions of the computer program;
- (3) updating the feedback data according to a pre-defined propagation scheme at multiple points during a compilation process;
- (4) performing an optimization of the first intermediate representation annotated with the feedback data updated in step (3) to produce a transformed intermediate representation; and
- (5) repeating steps (3) and (4) at least once.

OK  
SUBD<sup>1</sup>

15. The method of claim 14, wherein step (4) comprises the step of performing at least one of the following optimizations:

- (i) dead code elimination;
- (ii) dead store elimination;
- (iii) branch elimination; and
- (iv) code transformation.

16. The method of claim 14, wherein the first intermediate representation is a tree corresponding to a procedure within the source code.

17. The method of claim 14, wherein step (2) comprises the steps of:

- (a) constructing a control flow graph from the tree; and

(b) annotating a global or local frequency value of the global and local frequency data to an edge of the control flow graph, wherein the global or local frequency value corresponds to the number of times that the edge was traversed during the plurality of sample executions of the computer program.

18. The method of claim 17, wherein the global or local frequency value annotated to the edge of the control flow graph is one of the following:

- (i) EXACT;
- (ii) GUESS;
- (iii) UNKNOWN;
- (iv) UNINIT; and
- (v) ERROR.

19. The method of claim 17, wherein the global or local frequency value annotated to the edge of the control flow graph is one of the following:

- (i) GUESS; and
- (ii) UNKNOWN.

20. (New) A method for compile-time optimization comprising the steps of:

- (1) accessing a first intermediate representation of source code of a computer program, wherein the first intermediate representation includes instructions instrumented into the source code;

- Sub B57
- all
- (2) annotating the first intermediate representation with previously-gathered estimated frequency data from a plurality of sample executions of the computer program;
  - (3) updating the estimated frequency data according to a pre-defined propagation scheme;
  - (4) performing an optimization of the first intermediate representation annotated with the estimated frequency data updated in step (3) to produce a transformed intermediate representation; and
  - (5) repeating steps (3) and (4) at least once.
-